Code 582

Flight Software Systems Branch

FSB GUIDELINES FOR REQUIREMENTS MANAGEMENT IN MKS2007

Flight Software Systems Branch – Code 582

Version 1.0 – 07/23/08

582-2008-013

Goddard Space Flight Center

Greenbelt, Maryland

National Aeronautics and
Space Administration

UPDATE HISTORY

Version	Date	Description	Affected Pages
1.0	7/23/08	Baseline – K. Narayanan	All

CONTENTS

Introduction 1
1. Develop Requirements 1
2. Freeze Build Requirements and Conduct Build Start Review (for each Build) 2
3. Manage Related Development Tasks and Tests (for each Build) 2
Development 2
Test 3
4. Manage Requirement Changes 3
Change or Delete Draft Requirement (State = 'Propose') 3
Change Approved Requirement via DCR 3
Delete Approved Requirement via DCR 4
Add New Requirement via DCR 4
Update Requirements Document Version 4
Appendix – Requirement Workflow 5

Introduction

This document provides guidelines and procedures for requirements management in the MKS 2007 environment. See *MKS Integrity Manager 2007 User Guide* for additional information.

1. Develop Requirements

- Generate draft requirements directly in MKS.
  - Use *ISD Requirements Development Process*.
  - Use *FSB Requirements Document Template* to develop the up-front contextual part of the requirements document.
  - Open a new document using **Document > Create > Requirement Document**.
  - Enter new requirements in the **Document** view using **Content > Insert Content** menu. Assign unique internal requirement numbers. The initial ***State*** is '*Propose*.' Requirements may be revised indefinitely in this state, and no DCR will be needed.

    Note: Use internal requirement numbers – not MKS ID – for all requirement references. Do not modify the numbers of requirements from re-use library, if any.
  - Concentrate on requirements that need to be implemented in the next build – other requirements may remain in draft form (***State*** = '*Propose*'). Some requirements may be implemented in multiple builds.
  - Create Traceability links to parent and children requirements using '*Decomposed From/Decomposes To*' relationship.
  - Assign ***Planned Build***.
- To reuse requirements from another project:
  - Select the Requirements Document that will be reused, select **Document > Branch**, and select the name of the ***Project*** where the requirements will be used. This is done by the Requirements Admin.
  - The initial ***State*** of the requirements is '*Approved*' and any changes to requirements will require DCR.
- Conduct Requirements Peer Reviews as needed and incorporate recommendations into the requirements.
  - Generate a report from MKS for review. For reference include other previously approved requirements in MKS as well. Include ancillary information such as *Planned Build* and *State*.
  - Use *FSB Requirements Inspection Standard* to conduct the review.
  - Assign ***State*** as '*Approved*' for all new requirements that are approved. Any changes to approved requirements will require DCR.
  - Review/update ***Planned Build***.
  - Assign a version Label for the Requirements Document in MKS (e.g., Version 1.0) using **Document > Baseline**.

NOTE

To import initial requirements from **Word** or **Excel** documents into MKS contact the Requirements Admin. Once imported the requirements must be maintained and updated only in MKS.

2. Freeze Build Requirements and Conduct Build Start Review (for each Build)

- Assign ***Actual Build*** to new requirements in the Build. (Revised requirements should not change ***Actual Build***.)
- Generate a Build Requirements review document from MKS.
  - Include ***State, Planned Build, Actual Build***.
  - Include only current Build. However, it may be useful to review requirements planned for future Builds as well, so that some requirements could be reassigned if necessary.
  - Include already implemented requirements that have changed and need to be re-implemented in the Build (***State*** changed to '*Revision Approved'*).
  - Include DCRs on requirements that failed testing in previous build. More DCRs may be assigned to the Build during the Build development.
- Review the Build requirements with Build stakeholders – Test Team, Developers, and possibly others. Incorporate review comments into MKS.
- Generate Build Requirements Report from MKS – assign a version number and date, put it in CM.

NOTE

Ideally, Build development should start only after this review, but it may start earlier with the understanding that the delivered Build MUST implement the APPROVED requirements for that Build.

Similarly Test Team may develop scenarios and test procedures using draft requirements but the final procedures MUST verify APPROVED requirements.

3. Manage Related Development Tasks and Tests (for each Build)

Development

- Create a FSW *Implementation DCR* for each new development task for the Build. Link the DCR to the new requirements assigned to the Task for the Build, via '*Implements'* relationship.

  NOTE: Revised requirements are linked to other change DCRs via '*Related To*' relationship.
- Implement each Task. Link any new requirements that come in late to the DCR.
- When Build is ready for delivery assign ***Actual Build*** to the new requirements. Review all requirements for the Build and add, remove, or update their ***State***, ***Actual Build,*** and links as necessary.
- If not already done, update the version Label for the Requirements Document in MKS (e.g., Version 1.4, Build 2.0) using **Document > Baseline**.
- Create a Requirements report and/or Requirements Document report and attach it to the VDD. Also save it as an attachment with the Requirements Documents issue in MKS.

  NOTE: Requirements Document may not show which requirements are/are not implemented in a build. Another report with that information may be needed.

Test

- Create new *Tests* for the new Build. Link each Test to the applicable Build requirements via '*Validates'* relationship.

  NOTE: Typically several Tests are developed to test all requirements for a Task. Possibly multiple requirements are tested by each Test, and, some requirements may be tested by multiple Tests.
- Revise Tests for the previous Builds and their requirement links, as necessary.
- Before final Test execution all linked requirements should have the correct ***Actual Build*** – if not, update Test links and Tests.
- Execute Test. Each Test execution may pass some requirements and fail others. Set ***Build Test Status*** to 'Pass,' 'Fail,' 'Partial Pass' , or 'Untestable at this time' depending on Test result. Set ***Tested Build***.

  NOTE: Some requirements may be tested by multiple tests, and so ***Test Status*** should be the aggregate of all applicable tests.
- Submit Failed Test DCRs with links to Test and failed requirements.
- After Build Test is complete generate the RTTM and archive it with date.

NOTE

The requirements in MKS are dynamic and may be revised anytime. The fields, ***Planned Build, Actual Build, Revision Date and Revision Comment,*** along with the requirement ***State***, give a good indication of the requirements' development/revision status. (***Planned Build*** and ***Actual Build*** are never changed once the requirement is delivered in a Build.)

For each Build the Test Team must use the **version of the Build Requirements / Document** delivered along with the VDD for their tests. ***Tested Build*** and ***Build Test Status*** reflect the status of the **requirements as delivered in the Tested Build**. The Historic Reports generation feature in MKS may be used to view requirement changes since the build delivery.

4. Manage Requirement Changes

Change or Delete Draft Requirement (State = 'Propose')

- No DCR is necessary. Make changes as needed. Draft requirements changes/deletions are not counted for metrics.
- For deletions, add '**DELETED: '** before the requirement text and set Requirement ***State*** to '*Cancel*.' NOTE: Do not delete the requirement text.

Change Approved Requirement via DCR

- Link the DCR to the requirement if not already linked.
- Discuss at IRB and get approval – there may be impact to Development, or Test, or both.
- Set Requirement ***State*** to '*Revise'* and add ***Revision Comment***.
- Make the change.
- Get IRB approval and set Requirement ***State*** to '*Revision Approved*.'

- If change results in FSW change generate related FSW DCR and link it to the requirement.

  NOTE: Trivial typographical changes may be done without a DCR, after notifying IRB.

Delete Approved Requirement via DCR

- Link the DCR to the requirement if not already linked.
- Discuss at IRB and get approval.
- Set Requirement ***State*** to '*Revise'* and add ***Revision Comment***.
- Remove traceability links to high level requirements.
- Add '**DELETED: '** before the requirement text and set Requirement ***State*** to '*Cancel*.'

Add New Requirement via DCR

- Discuss DCR at IRB and get approval.
- Create the requirement in the **Document** view using **Content > Insert Content** menu.
- Link the requirement to the DCR. Get IRB approval.
- Set Requirement ***State*** to '*Approved*.' Assign ***Planned Build***. Link it to the Build implementation DCR.

Update Requirements Document Version

- If there were changes to requirements or document, before next delivery use **Document > Baseline** to update the version Label for the Requirements Document in MKS (e.g., Version 1.4).

Appendix – Requirement Workflow

This appendix shows the requirement workflow.

- Requirement states – *Propose*, *Approved*, *Revise*, *Revision Approved*, and *Cancel*. See Figure 1.

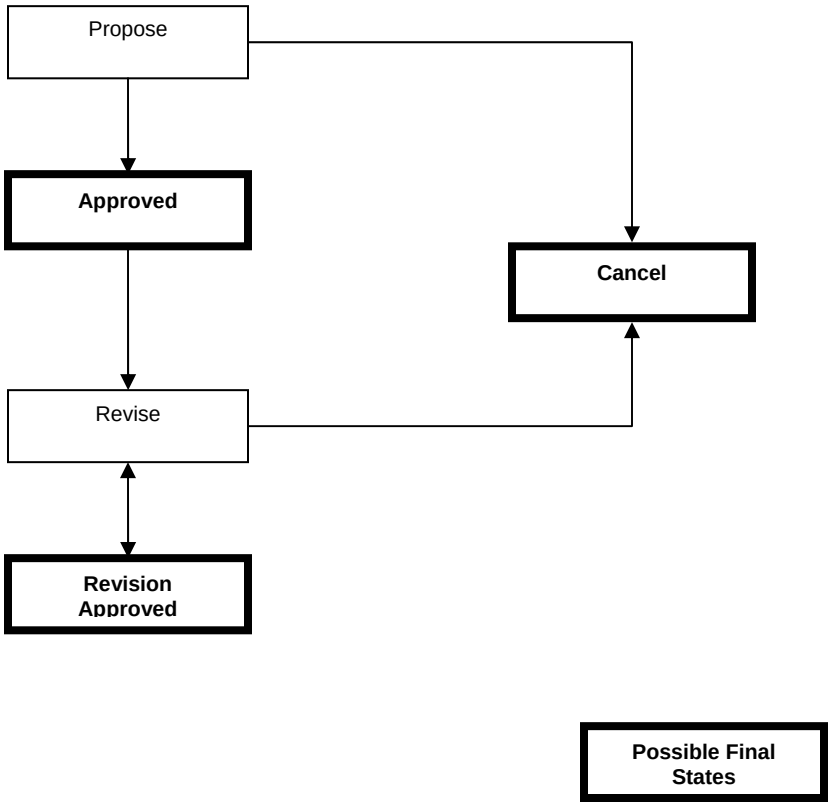

Figure 1. Requirement Work Flow